School of Computing and Information Systems
The University of Melbourne
COMP30018/COMP90049
Knowledge Technologies (Semester 1, 2019)
Workshop sample solutions: Week 2

1. What do we mean when we say **knowledge technologies**?

   (a) Revise the definition of **knowledge tasks** (or **complicated** problems), with respect to **concrete tasks** (or **simple** problems).

   - From the lecture slides: concrete tasks are well-defined, we can assess whether the solution to the problem is "correct," the data is transformed in a mechanical way, and there is limited contribution to human knowledge; for knowledge tasks, the outcome is not well-defined, computers are used to mediate between the users and the data, and context (provided by the user) is critical.
   - Because all tasks are ultimately mechanical for computers, any task can be construed as a concrete task in this context. On the other hand, any problem can eventually form part of a larger system which can lead to human knowledge, so any task can potentially be a knowledge technology. To be able to decide, we must provide a context for the problem — in particular, in terms of the **user** and how the solution might extend their knowledge.

   (b) Consider the following, and decide into which category you believe they fall, referring to the definition you have decided upon above.

   - The classification depends on how your define the user's context (or lack thereof); for just about all of these, the description is too vague to be completely certain whether or not there is a knowledge component. In a sense, they could be either, depending on how we perceive the role of the user, what knowledge they can gain, and whether different users have different solutions.

   i. Multiplying two floating-point numbers in base 16

   ii. Playing a competitive game of naughts-and-crosses

   iii. Playing a competitive game of go

   - Recently, certain computer programs have been shown to be as good as (or better!) than the best human players of go.
   - In some sense, this seems like it might have a concrete objective — whether the computer program has won or lost is not subjective — but the "best" sequence of moves is clearly still unknown. The current computer programs are clearly just following a mechanical set of operations to beat the best human players, but there isn't a sequence of operations that we can follow to definitively determine whether there exists other strategies that are better yet. This seems to suggest that this is a knowledge task.
   - In terms of contributing to human understanding, one striking remark by the professionals is how much they learned from the go engine — that they thought they already knew the best ways to play, but then the computer showed them that there were better ways that humans had never tried.

   iv. Playing a competitive game of tennis

   v. Calculating the trajectory of a thrown book

   vi. Selecting appropriate counter-measures after someone has thrown a book at you

   vii. Selecting a book that a given person will enjoy reading

   viii. Translating a program written in C into Java

   - If we provide some context along the lines of "an accurate translation of the program would cause the computer to enter the same states, and produce the same output," then we would probably have a concrete task — although it might not be clear exactly how to do this, or whether there are multiple translations that achieve the same solution, it is certainly the case that a set of observers would not disagree: either the program enters the same states, or it does not. There is no contextual interpretation of this goal.

- On the other hand, we aren't told *why* we want such a translation. By introducing a user, we might actually have a knowledge task. For example, if the purpose of the translation is to help a C expert to learn how to write Java programs, then the mechanism of the translation is not so clear any more. People learn in different ways; simply producing the same output with an inordinately complex syntax doesn't guarantee our true goal of having the user learn Java. The "best" way to translate the C program depends on the user's context (what they already know about C and Java, and how they acquire knew information), and so this would instead be a knowledge task.

ix. Translating a document written in Japanese in English

2. How is **data** different to **knowledge**?

   - In extension from the above definition, data is just the various sensor readings/variables/record values that characterise some system, whereas knowledge results from interpreting the data in a way that helps the user to solve their problem (and learn something).

3. Describe a process through which we might be able to answer the question "Where shall we go for dinner tonight?" using Google (`http://www.google.com`) as a resource.
   (We'll touch on some of these elements as the semester goes on.)

   - Conceptually, we want to construct a query out of good keywords, parse the results that Google gives into web pages, read the web pages to find restaurants which meet our criteria (whatever those are), and then choose a single restaurant to be the response of the system. (This single response would provide the knowledge; everything else is just data!)

4. Revise the following **regular expression** operators:

   `( ) [ ] { } . * + ? ^ $ | \`

   For each of the following, give a couple of examples of strings which the regular expression would match. Describe (colloquially, in a manner that a non-technical person would understand) the set of strings that the pattern is designed to match.

   (a) `/[a-zA-Z]+/`

   - Strings that contain one or more (English) letter characters, e.g. `cat`, `555Z29E8!`

   (b) `/^[A-Za-z][a-z]*$/`

   - Strings that consist entirely of one or more (English) letter characters, with the first character possibly uppercase, e.g. `Melbourne`, `ff`

   (c) `/p[aeiou]{,2}t/`

   - Strings that contain the letters `p` and `t`, with up to two (English) vowels in between them, e.g. `peat`, `aptitude`

   (d) `/\s(\w+)\s\1/`

   - Strings that contain a repeated "word" (set of alphanumeric characters separated by whitespace), e.g. `0␣0␣0`, `\ncat\ncattle\n`